Scandinavian Journal of Information Systems

Volume 33 | Issue 1 Article 5

6-30-2021

Platformizing the Organization through Decoupling and Recoupling: A longitudinal case study of a government agency

Kathrine Vestues
Norwegian university of science and technology, kathrine.vestues@ntnu.no

Knut Rolland
University of Oslo, knutr@ifi.uio.no

Platformizing the Organization through Decoupling and Recoupling

A longitudinal case study of a government agency

Kathrine Vestues
Department of Computer and Information Sciences, Norwegian University of Science and Technology
kathrine.vestues@ntnu.no

Knut H. Rolland
Department of Informatics, University of Oslo; University College Kristiania
knutro@ifi.uio.no

Abstract. Digital platforms are known for their ability to produce and reproduce flexible solutions for multiple user groups and for attracting third-party developers. Consequently, the unit of analysis for digital platforms is often the consumer market. In contrast, in this paper, we investigate how platformization has the potential to increase digital innovation within organizations through a transformation of legacy systems and the organizational structure. Empirically, we draw on a longitudinal case study of a Norwegian government agency following their platformization process over two years. In advancing a sociotechnical perspective on platformization, we offer three distinct contributions. First, we contribute by theorizing how existing legacy systems are discontinued in processes of decoupling. Second, we capture how this enables novel recombination of knowledge and skills in the recoupling processes, which, in turn, facilitates new ways of working and organizing. Third, we theorize on how decoupling and recoupling processes interact, thus facilitating increased levels of platformization.

Key words: Platformization, Digital platforms, Digital infrastructures, Digital transformation, IS development.

Accepting editor: Katrin Jonsson

1 Introduction

Digital platforms have attracted considerable attention from practitioners and information systems (IS) scholars. For practitioners, digital platforms have become the newest buzzword for establishing new business and rapidly scaling services to a large number of consumers. In the IS literature, digital platforms are often portrayed as multi-sided markets enabled by flexible platform architectures. As a combination of technical architectures and business models, digital platforms are capable of producing very varied, as well as high-quality, products and services (Cennamo and Santaló 2019; Gawer and Cusumano 2014; Parker et al. 2016; Tiwana 2010; Wareham et al. 2014; Yoo et al. 2010). Thus, digital platforms are often put forward as technologies with an almost inherent capacity for continuously producing new features and innovations. For example, IS researchers recently investigated how innovation platforms, such as Google Android and Apple iOS, evolve through the provisioning of 'boundary resources' that balance platform control while retaining flexibility and openness to attract third-party developers (Eaton et al. 2015; Ghazawneh and Henfridsson 2013).

Additionally, although not frequently investigated, many organizations utilize platforms as part of the organizations' digital infrastructure—especially as a more interoperable and flexible replacement for existing legacy systems (Rolland et al. 2018). Thus, the current body of research on digital infrastructures is relevant for understanding how organizations adopt and adapt digital platforms, and in particular, how organizations struggle with the inertia of an installed base (Aanestad and Jensen 2011). Therefore, solving problems related to the inertia of the installed base has been a long-standing concern among Scandinavian IS scholars. This stream of IS research has emphasized various solutions and conceptualizations, for example, *cultivation* (Aanestad and Jensen 2011; Ciborra 2000), *gateways* (Hanseth 2001), and *bootstrapping* (Hanseth and Lyytinen 2010). To this end, utilizing platform architectures and design principles seems highly relevant for redesigning infrastructures into modularized architectures that facilitate change and modifications.

While building and expanding on these insights, we investigate a different strategy based on platformization of digital infrastructures in organizations. Platformization has been suggested as a viable way of increasing flexibility and innovation in digital infrastructures (Bygstad and Hanseth 2018; Islind et al. 2016). By breaking monolithic systems into modular applications, functionality can be added, changed, or removed without affecting other applications and modules. For most organizations, however, this strategy requires a gradual transition, in which legacy systems are gradually dismantled into modular applications (Bygstad and Hanseth 2018). Although digital platforms come in many technical forms and organizational arrangements (Gawer and

Cusumano 2014), their layered modular architecture (Rolland et al. 2018; Yoo et al. 2010), and the way in which boundary resources within the platforms can be combined and recombined to produce novel solutions and products (Constantinides et al. 2018; de Reuver et al. 2017; Henfridsson and Bygstad 2013), make them especially suited for transforming existing infrastructures and improving the capabilities for innovation and change.

Against this backdrop, our objective is to explore the potential for digital platforms to allow organizations to change the way they develop and maintain software and thus, increase flexibility and innovation. The mediating capacity of platforms has enabled new ways of developing and offering services (Rolland et al. 2018). However, this capacity is not restricted to open platforms and external developers: By using platforms internally to coordinate the efforts of autonomous teams, organizations can increase flexibility and innovation within the boundaries of a firm. The modular layered architecture of the platform (Yoo et al. 2010) allows cross-functional teams to develop and deploy applications without coordinating their efforts with other teams and applications. This potentially enables innovation which is difficult in fragmented digital infrastructures consisting of a web of interconnected legacy systems. Based on the transformative potential of internal digital platforms, we ask the following research question

> *What characterizes how processes of platformization unfold in large-scale organizations?*

We explored this question through a longitudinal case study within NAV, the Norwegian Labor and Welfare Administration. NAV is the backbone of the Norwegian welfare system and is responsible for redistributing one third of the national budget through programs such as age pensions, sick leave benefits, and disability benefits. The agency is also responsible for stimulating the population's work ability and increasing the number of citizens in active employment.

In 2017, NAV embarked on a major platformization of its digital infrastructure, in which legacy systems were dismantled into modular applications. The present study results show that the dismantling of monolithic systems enables radical changes in the way organizations develop and deliver digital services. Theoretically, we conceptualize platformization as the unfolding of two interrelated processes of decoupling and recoupling. These concepts reflect insights from service-dominant logic, which sees innovation as unfolding through the related activities of liquefaction and resource integration (Lusch and Nambisan 2015). Decoupling denotes the dismantling of legacy systems into modular applications by, for example, using so-called microservices, Platform as a Service (PaaS), and container platforms. Recoupling refers to the social changes enabled by the decoupling. Thus, platformization during decoupling implies utilizing and introducing a platform-oriented governance model combined with the loosely coupled architecture of a platform core and a wide variety of different modules (apps) running on top. By decoupling legacy systems, the organization increases the availability of the digital components. These components can, in turn, be combined and recombined to produce novel services. Recoupling establishes roles and practices that facilitate recombination—providing flexibility and innovation.

Our contribution is threefold. First, we contribute by theorizing how existing legacy systems are discontinued in decoupling processes. Second, we capture how this enables novel recombination of knowledge and skills in recoupling processes, which, in turn, facilitates new ways of working and organizing. Third, we theorize on how decoupling and recoupling processes interact, thus facilitating increased levels of platformization. Generally, decoupling and recoupling processes, although distinctly different, are mutually interdependent. Decoupling paves the way for recoupling, and in turn, recoupling increases the organizational capabilities for undertaking further decoupling. Thus, this theorizing makes the fundamental assumption that platformization is a sociotechnical phenomenon that to succeed requires digital platforms and the appropriate organizational processes and capabilities.

The remainder of the paper is organized as follows. First, we introduce related literature on digital platforms, before we present a theoretical framework for platformization. We then formulate the research methods and results. Finally, we discuss the findings and make concluding remarks.

2 Digital platforms in organizations

In the past decade, the body of scholarly research on digital platforms has grown considerably. IS scholars have published numerous works, including theories of how platforms evolve (Eaton et al. 2015; Ghazawneh and Henfridsson 2013), how platforms increase flexibility in organizations (Rolland et al. 2018), and how platforms enable innovation based on modular layered architectures (Yoo et al. 2010). Platforms and ecosystems have been studied from multiple perspectives and across different fields of research. For example, scholars from disciplines such as economics (Parker et al. 2016; Tan and Wright 2018), accounting (Kornberger et al. 2017), strategy (Jacobides et al. 2018), and systems engineering (Boudreau 2010; Spagnoletti et al. 2015; Tiwana et al. 2010) have conducted empirical studies of platforms.

In a review of the platform literature, Gawer (2014) argues that the discourse on digital platforms has been dominated by two distinct theoretical perspectives: one inspired by industrial economics, the other by engineering design. From an economic perspective, platforms create value by acting as mediators between two or more categories of consumers who would otherwise not connect. This value is linked to platforms as multi-sided markets that produce network effects (Parker et. al 2016). Network effects relate to the ability to build networks where any additional user will enhance the experience of existing users (Vassilakopoulou et al. 2017). The more users participate in the network, the more valuable the network becomes for each participant.

However, the economic perspective has several limitations. First, it sees the platform as fixed and exogenous, and does not consider how platforms evolve and interact with their surroundings. Similarly, as noted by de Reuver et al. (2017), the lack of emphasis on the technical architecture means that many studies within this research stream fail to conceptualize the digital features and affordances of platforms. Certainly, various platforms can be defined as multi-sided markets but will still differ in terms of digital architectures. This is important for how a platform may evolve—or fail to evolve (Rolland et al. 2018). Second, the interaction between the owner and the complementor is reduced to a simple producer-consumer relationship, which fails to capture the complex relationship between different actors in the ecosystem. In platform ecosystems, an actor consumes and produces services, sometimes as part of the same transaction (Lusch and Nambisan 2015). Different technical configurations also affect the way in which actors co-create value. Based on these shortcomings, Gawer (2014) concludes that although the economic perspective provides valuable perspectives on the platform's ability to mediate and coordinate, this perspective fails to address concerns related to innovation and evolution.

The second perspective is the engineering perspective, which sees platforms as technical artifacts with a modular architecture that have a stable core and numerous peripheral components. For instance, Baldwin and Woodward (2009) argue that all platforms share a common trait: They have a modular architecture that is centered on the core and the periphery. In this view, "a platform architecture partitions a system into stable core components and variable peripheral components" (ibid., 24). The layered modular architecture makes platforms particularly well suited for innovation. The reason is three-fold. First, modularity enables the recombination of modules (Henfridsson et al. 2018). Recombination is the most basic form of innovation and happens as actors combine existing modules into novel products or services. By breaking a system into smaller parts with standardized interfaces, modules can more easily be recombined.

Second, defining clear interfaces between modules allows for specialization and autonomous innovation. As long as interfaces remain intact, a development team can innovate within the confines of a module, without affecting other teams or modules. Third, the separation between the core and the periphery abstracts the complexity of the underlying infrastructure, providing developers with "valuable ignorance" of the core's native functionalities (Tiwana 2013, p. 82). This increases the speed of innovation as developers can focus on their own work yet be able to integrate their applications with the platform (ibid.).

However, the engineering perspective also has limitations. First, platforms are viewed as relatively stable structures in terms of software architecture and apps with limited attention in the way in which they evolve. Within organizations, however, there are usually intersecting projects, diverging needs, and numerous legacy systems that will influence and shape the evolution of the platform. Second, although the engineering perspective provides insight into the platform's ability to facilitate distributed and autonomous innovation, the literature provides few insights into how autonomy is exercised and how complementors innovate. Third, from the engineering perspective, innovation is seen as unbounded (Yoo et al. 2012)—limited only by the availability of components. Platforms introduced as a means for increased innovation within organizations, however, are limited by economic, structural, cognitive, and institutional constraints (Yoo et al. 2010, 730).

Based on the identified shortcomings, Gawer (2014) proposes a third perspective on digital platforms that bridges the gap between the economic and engineering perspectives. This third perspective is described as "organizational" and sees platforms as evolving organizations or meta-organizations that "federate and coordinate constitutive agents who can innovate and compete," and technical structures with "a modular technological architecture composed of a core and a periphery." In contrast to the economic and engineering perspectives, the organizational perspective is explicitly sociotechnical, conceptualizing platforms as a "sociotechnical assemblage encompassing the technical elements (of software and hardware) and associated organizational processes and standards" (de Reuver et al. 2017, p. 126). The organizational perspective differs from the engineering perspective in its view on innovation: The engineering perspective sees platform innovation as unbounded, but the organizational view acknowledges that the innovative capacity is influenced by organizational structures and practices.

However, although the organizational research stream acknowledges the bounded, sociotechnical, and evolutionary characteristics of digital platforms, it has several limitations. First, most extant studies adopt an owner-centric view, focusing on the way that platform owners design platforms to maximize their own profits (Spagnoletti et al. 2015). Thus, the organizational stream gives limited attention to the challenges and opportunities that digital platforms give user organizations. Without insight into the way end-users and their organizations develop and use digital platforms, it is difficult to understand how different configurations of platforms and user organizations might enhance or impede innovation. Second, the owner-centric focus gives little attention to the way in which legacy systems affect platform evolution. Today's business organizations often utilize and build on many different platforms and cloud services as an integral part of their infrastructure. For example, Rolland et al. (2018) report on a longitudinal case study of a digital platform for news production in a media company. In this study, the digital platform provides novel opportunities for expanding the company's digital infrastructure and supporting new services and ways of organizing journalists' work. At the same time, this research shows the challenging dilemmas and the balancing act between managing the inertia of legacy systems and the vast opportunities provided by the platform ecosystem. Third, although the organizational stream emphasizes the evolutionary characteristics of digital platforms, it pays limited attention to how these platforms come into existence. Most extant studies examine the evolution of established platforms (Eaton et al. 2015).

In response to these shortcomings, a growing number of studies are adopting a platformization perspective (Bygstad and Hanseth 2018; Islind et al. 2016; Poell et al. 2019; Törmer and Henningsson 2018), examining the way in which digital platforms are created, how they evolve, and the way in which they interact with existing infrastructures.

3 Theorizing platformization

Platformization has become a popular strategy for breaking away from a drifting digital infrastructure (Törmer and Henningsson 2018) and making the digital infrastructure more flexible (Battleson et al. 2016). Helmond (2015) uses the term to refer to the rise of the platform as the dominant infrastructural and economic model of the social web, while Islind et al. (2016) describe platformization as the socio-technical process of creating a platform. Bygstad and Hanseth (2018) define platformization as a process where silo solutions are decoupled into a platform-oriented infrastructure.

In this paper, we also see platformization as the decoupling of systems into a platform-oriented structure. However, we define platformization not only as the process of picking apart but also as the process of putting back together. This implies that platformization consists of two basic processes. The first process decouples systems, information, and activities into modular components, and the second process comes from being able to recouple components that would otherwise be difficult or expensive to combine (Normann 2001).

These insights are compatible with service-dominant logic (Lusch and Nambisan 2015), where innovation is seen to happen through the interrelated processes of 'resource liquefaction' and 'resource integration.' From this perspective, resource liquefaction is defined as the decoupling of information from its physical media. As information is decoupled, it becomes a resource that can more easily be transmitted and recombined by other actors. The availability of resources is measured in terms of 'density,' where maximum density occurs when the best combination of resources is mobilized for a particular context (Normann 2001). Value is created and co-created through the integration of resources.

However, while service-dominant logic sees resource density as the decoupling of information, we take a broader view and include the decoupling of systems. The reason is related to the accessibility of information stored in legacy systems: Although the information is accessible in principle (Kallinikos et al. 2013), poorly designed interfaces combined with tightly coupled systems in many cases make information difficult or impossible to access and modify. To increase the availability of information for external modification and use, legacy systems should be decoupled into smaller components with clearly defined interfaces. For organizations with a large legacy of information systems, resource density can be increased in two ways: either through the decoupling of information from its physical media or through the decoupling of legacy systems into modular components.

Digital platforms with their layered modular structure are particularly well suited for increasing resource density and facilitating resource integration (Yoo et al. 2012). This is because digital platforms are characterized by standardized interfaces between components—making them easy to access and combine within and across layers. The platform also provides a venue where actors can access and exchange resources, thus enabling resource integration and innovation.

However, the platform structure with its stable core and modular periphery not only leverages resource density and enhances resource integration. The platform also provides a fluid structure that enables a dynamic reconfiguration of value paths. As work processes are decoupled into modular applications, the processes can be reconfigured and reallocated to the most appropriate part of the organization. This allows for a dynamic reconfiguration of work practices and organizational arrangement by interleaving the decoupling and recoupling processes. Novel value paths will emerge—gradually transforming the organization. Eric Evans (2006) describes the challenges involved in achieving such modularity:

> The goal of the most ambitious enterprise system is a tightly interconnected system spanning the entire business. Yet the entire business model for almost any such organization is too large and complex to manage or even understand as a single unit. The system must be broken into smaller parts, in both concept and implementation. The challenge is to accomplish this modularity without losing the benefits of integration, allowing different parts of the system to interoperate to support the coordination of various business operations.

Concepts	*Activities involved*	*References*
Decoupling	Decoupling information from its physical media (Liquefaction)	(Normann 2001; Lusch and Nambisan 2015)
	Decoupling legacy systems into modular applications	(Bygstad and Hanseth 2018)
Recoupling	Recombining knowledge and skills into new organizational forms and practices	(Fitzgerald and Stol 2017; Brown and Duguid 1991)
	Recombining applications and teams into domains	(Evans 2006; Zammuto et al. 2007)

Table 1. Platformization theorized.

4 Research method

4.1 Case background

Fieldwork was conducted within the IT department at NAV. NAV was established in 2006 following reform of the Norwegian welfare system. The reform involved a merger of the formerly separate Employment Services and National Insurance Services. In addition, the reform involved a formal collaboration between NAV and municipal social services. In total, the organization employs approximately 19,000 people, of whom 14,000 are central government employees, and 5,000 are municipality employees.

NAV has a dual responsibility: The agency stimulates the population's work ability to increase the number of citizens in active employment and supports them economically during periods when they are unable to support themselves. NAV redistributes close to one third of the Norwegian national budget through programs such as unemployment benefits, age pensions, and sick leave benefits. NAV forms the backbone of the Norwegian welfare state, and most Norwegian citizens have contact with the agency at some point.

The NAV IT department develops, maintains, and operates the information systems that support the organization. The IT department has approximately 700 employees and 400 consultants, and maintains and operates close to 300 applications. The application portfolio is made up of several generations of solutions; from mainframe systems to newer web-oriented applications, as well as standard systems that support operations such as accounting, payroll, and document production. Efforts to increase efficiency and automation have made applications increasingly interconnected. Historically, these dependencies have been addressed through centralized control and staged deliveries, and software development has been outsourced to external suppliers. However, although the number of errors has been reduced, the delivery method has proven inflexible and expensive.

Therefore, in 2017, the organization changed its development and sourcing strategy. To increase flexibility and reduce costs, NAV moved from centralized control and outsourcing of software development to internal ownership and a distributed governance model. This study focuses on how this transition was enabled by changes in the digital infrastructure and organizational structure through the process of platformization.

4.2 Data collection

Most data were collected by the first author between March 2017 and May 2019. We used three methods to collect data: interviews, participant observations, and document analysis. First, we conducted a total of 38 interviews (see Table 2). Of these interviews, 23 were recorded and transcribed. The interviews lasted between 45 and 60 minutes. Because of their sensitive nature, not all interviews could be recorded. In these cases, we took notes and added more extensive remarks shortly after the interviews ended.

Informants were chosen from across the IT department based on a snowballing strategy, where one informant suggested the next. Most informants worked on a large ongoing modernization project (the Parental Benefit project) that developed software

Roles	*IT architecture section*	*Parental Benefit project*	*Other sections*	*Total*
Senior executives	1	2	3	6
Project managers		4		4
Team leads		4	2	6
Developers		8	4	12
Architects	3	4	1	8
Case workers			2	2
Total	4	22	12	38

Table 2. Overview of interviews

support for improved management of parental benefits. Parental benefit is a welfare benefit intended to compensate parents for loss of income in relation to the birth or adoption of a child. Parental benefits payments were administered on an IBM mainframe solution introduced in 1978, and the project was a step toward replacing the old system with more flexible and integrated systems support. With an estimated cost of close to 1.3 billion NOK, the Parental Benefit project was the largest ongoing development project within the Norwegian public sector at the time of data collection and provided unique insight into the challenges of replacing legacy systems. The project also gave us insight into the transition from staged to continuous development practices in large-scale organizational settings. In addition to the Parental Benefit project, several informants worked in the section for the IT architecture section. The IT architecture function in NAV has traditionally been responsible for ensuring consistency and sustainability across projects and has had considerable influence on decisions related to technology and architecture. With the transition from a centralized to a distributed decision model, the IT architects were assigned to development teams, consequently losing some of their influence over technology and architecture decisions.

Another important data source was participant observations. The first author was able to move freely within the IT department. She could also attend most meetings and social gatherings. Considerable insight was gained through informal conversations by the coffee machine and encounters in the hallway. Observations and conversations were documented extensively in a field diary.

The data also included a significant volume of documents collected from internal and external websites and archiving systems. Among these documents were government white papers, procurement documents, design specifications, project reports, and project websites. The first author was given an e-mail address and had access to most internal documents, including calendars, project wikis (Confluence), and issue tracking systems (JIRA). In addition, online conference presentations by NAV employees were transcribed and analyzed. The transformation of NAV has attracted considerable attention in the software development community, and NAV has presented the change process at a series of practitioner conferences in Norway.

4.3 Data analysis

Data analysis was iterative and overlapped with data collection, thus granting flexibility to respond to emergent themes (Eisenhardt 1989). Specifically, the data analysis can be described as an iterative four-step process. First, we deductively started from a platformization framework derived from the platform literature. This framework indicated that platformization consisted of two separate but related processes: One involved dismantling legacy systems (Bygstad and Hanseth 2018), and the other involved an altered organizing logic (Yoo et al. 2010). We labeled the first process 'decoupling,' and the second 'recoupling,' where decoupling referred to the dismantling of systems, and recoupling referred to the associated organizational changes.

Second, we used an open coding procedure to discover concepts and their properties and dimensions (Charmaz 2014). We manually coded the data using colors and annotations. We developed descriptive codes capturing the informants' views and reflections on the organizational and technological changes leading up to the transition. Following the actor-centric principle of interpretive research, we identified what was perceived as a problem, by whom it was perceived as a problem, and solutions proposed. For instance, an informant described the problems involved in operating several technical platforms simultaneously. He suggested that these problems had been solved through standardization on a single application platform, separating the application layer from its underlying technical resources. From this, we arrived at the code 'Separating technical infrastructure from the presentation layer.' Similarly, an informant described how tightly coupled systems required extensive coordination between projects. This was, according to the informant, addressed by dismantling legacy systems into modular components that could be developed and managed independently. From this, we derived the code 'Legacy systems are dismantled into smaller components.'

Third, we merged the descriptive codes with the theoretical constructs derived in the first step of the analysis by identifying empirical examples of decoupling and recoupling, thus describing how these processes unfolded in the organization. Through discussions with colleagues, we were made aware of the similarity between our conceptualization of decoupling and recoupling and the concepts resource liquefaction and resource integration found in service-dominant logic (Lusch and Nambisan 2015). This insight was fed into the analysis, thus triggering an understanding of decoupling as a process of picking apart, and recoupling as a process of putting the pieces back together. A mapping between theoretical constructs (decoupling and recoupling) and their associated descriptive codes is given in Table 3.

5 Results

In the following, we present the results of the study. We begin by describing the process of decoupling, before describing how recoupling unfolded in the study. Figure 1 provides a timeline of the most important events identified in relation to each process.

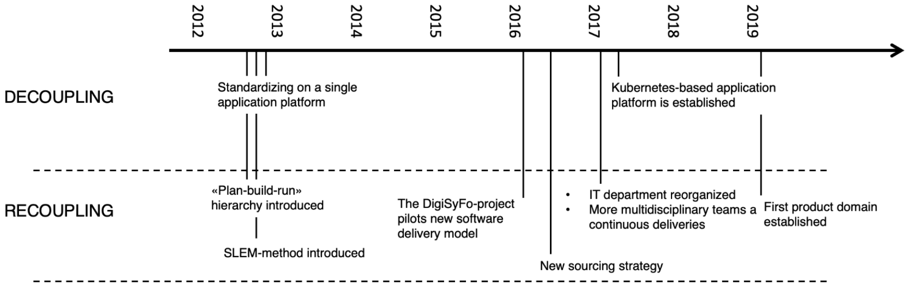
2012
2013
2014
2015
2016
2017
2018
2019
DECOUPLING
Standardizing on a single application platform
Kubernetes-based application platform is established
RECOUPLING
«Plan-build-run» hierarchy introduced
SLEM-method introduced
The DigiSyFo-project pilots new software delivery model
New sourcing strategy
• IT department reorganized
• More multidisciplinary teams a continuous deliveries
First product domain established

Figure 1. Events relevant to the platformization processes in NAV.

Constructs	*Descriptive code*	*Excerpts*
Decoupling	Decoupling between technical infrastructure and presentation layer	"We abstracted away some of the complexity by saying that if you are going to build an application, then you should not have to think about which operating system and network protocols to use, and low-level technical things." (Architect IT, section for IT architecture)
	Decoupling of legacy systems into smaller components	"Systems should be broken up into products then that can be developed, tested and operated separately. Rather than one big lump, you need several separate components. In this way, you avoid all the coordination which takes a lot of time. If my component is dependent on your component, we have to make sure that your component is released before my components and so on. This is very demanding." (Software developer, Parental benefit project)
Recoupling	Establishing independent teams	"This project aimed to develop digital solutions for sick leave services targeting employees, employers, health personnel and NAV employees managing the services. The project was organized in two teams and it was decided to give the teams 'autonomy'; defined as giving the team members freedom to work in close cooperation with product owners and users to decide the scope of the work, to take ownership of their processes and practices, and to take responsibility for the interfaces with other systems in a proactive manner." (Team member from the first independent team).
	Establishing product domains	"The idea behind domain-driven design is that it is much more important to organize for flow: Optimal flow in your organization and in your code. And you do this by minimizing coordination—chattiness across [the organization]. You want to cluster things that naturally belong together and keep things that do not belong together apart. You cluster people and code into domains and domain areas. Then you get a closer connection between people within a domain." (Senior executive)

Table 3. Examples of theoretical constructs and associated codes

5.1 Decoupling legacy systems (2012-present)

The digital infrastructure was decoupled in two stages: First, the organization was standardized on hardware and middleware through the introduction of a common application platform. Second, legacy systems were rewritten as modular applications.

First decoupling phase: Establishing a platform core. The first phase began in 2012 when NAV introduced the agency's first application platform. Previously, each project had been responsible for specifying and establishing its own technical infrastructure, resulting in a multitude of different technological platforms. Technical heterogeneity, combined with manual deployment and provisioning, made application management error-prone and time-consuming. To reduce error rates and increase efficiency, NAV decided to standardize on a single platform and streamline operations management. The application platform, which was named 'the Cloud,' was based on virtual server technology, where JBoss and WebSphere middleware ran on a Red Hat Linux operating system. Software development projects were instructed to use the platform, thus reducing heterogeneity and simplifying deployment and provisioning. The introduction of virtual server technology reduced hardware costs, and semi-automated provisioning reduced the setup time from weeks to minutes. By standardizing on a single application platform, projects no longer had to spend resources on choosing and setting up technical infrastructure.

> Let me give you an example. In the first project I worked in - it must have been in 2006, we used a lot of time, not only to decide which application platform to use but also choosing the operating system it would run on. A project destined to build a case tool would spend a lot of time deciding whether the application should run on Windows or Linux, or if it should run on a mainframe solution. We spent a lot of time doing this. And projects don't spend time doing this anymore. It's been standardized. Everyone just has to relate to it (Architect in IT architecture section).

Most systems were migrated to the application platform, and by the end of 2015, the plethora of hardware and middleware technologies had been reduced to three technical platforms: 1) the application platform (the Cloud); 2) Arena, an Oracle forms-based system for managing follow-up activities related to employment; and 3) InfoTrygd, an IBM mainframe system from 1978 used to manage individual benefits.

However, although the platform abstracted the underlying complexity and simplified provisioning and deployment, the applications were still large and interconnected.

To manage this complexity, the organization centralized operations and banned projects from releasing their own applications. Deployments were restricted to four yearly releases, where all changes had to be approved and deployed by the Operations department. A single release could include 80 000 development hours.

> In 2014, we had a major delivery of approximately 80 000 development hours. [...] it's the biggest we've had. Most of them are around 40-50 000 (Project manager, NAV).

Coordinated releases and centralized control reduced the number of errors, but it could be months from the time a feature was developed until it was available to users and development teams. It also increased complexity as developers were forced to anticipate needs, and "think about everything" upfront.

Second decoupling phase: Establishing a variable periphery of applications. To reduce the complexity of the digital infrastructure, in 2017 NAV began to decouple monolithic systems into modular applications. To facilitate the process, the organization introduced a second application platform. The application platform was based on Kubernetes, an open-source framework for managing software containers. In this way, NAV was able to design its own platform, offering only a subset of the functionality available on the Kubernetes platform. A software container has its own filesystem, CPU, memory, and process space, and as they are decoupled from the underlying infrastructure, they are portable across clouds and operating systems. By containerizing applications, they could easily be migrated between platforms.

As part of the migration process, large systems were also broken down into smaller applications— thus reducing complexity. The modularized architecture made it possible to deploy and manage applications independently. This was also one of the main goals for introducing the new application platform: to introduce a distributed governance model where each team could deploy and manage their own applications. In this way, applications could be developed and released at the discretion of the team, thus increasing the rate of change. With the new governance model, release rates were increased from once every few months to several times a day. As release rates increased, so did the feedback and development speed. One manager described the transition from the old to the new structure as "taking a super-tanker and splitting it up into 100 speedboats" (Section manager in the IT department).

An important prerequisite for the distributed governance was automation of operations management: The platform provided fully automated services for tasks such as provisioning, deployment, and load balancing. This meant that applications could be managed without expert knowledge in operations management, enabling teams to take responsibility for their own applications. A team lead for the platform team stated:

> Kubernetes is the open-source framework that comes from Google. It is all of Google's experience over the last 15 years with how to manage infrastructure rewritten by the same people. It's like taking the world's best operations person and fully automating him. That's what Kubernetes is. It provides a lot of tools for running in production, which makes it more robust, and more scalable and everything.

To ensure that the transition from centralized to decentralized governance did not result in chaos, NAV introduced the concept 'white-listing.' Applications were white-listed if they were sufficiently low in complexity and had few external dependencies. Only white-listed applications could be managed by the development teams. Complex applications still had to be managed by the Operations department.

5.2 Recoupling the organization (2016-present)

The recoupling of NAV can be seen to progress in two overlapping stages: In the first stage (2016 and onward), NAV introduced a distributed service delivery model, where multidisciplinary teams began to develop and maintain their own applications. In the second stage, teams and applications were recombined into product domains.

First recoupling phase: Establishing multidisciplinary teams. In 2016, NAV initiated a project that would pilot a new software delivery model. The project, which we called DigiSyFO (short in Norwegian for Digital sick leave follow-up), was responsible for digitizing follow-up activities related to sick leave. The project had two goals: First, the project would digitize follow-up, transferring communication between the employer, general practitioners, and NAV from analogue to digital media. Second, the project would pilot a new software delivery model, where staged deliveries and centralized control were replaced by multidisciplinary teams that were responsible for the entire software delivery cycle. Software would be developed and released continuously. Whereas other projects employed a staged delivery model with a handover between stages, the Digital Health project took responsibility for all parts of the development cycle, interleaving activities such as design, development, and operations.

> The project was initially set up with a single team with less than ten members consisting of two UX designers, two subject matter experts, one software architect, one developer, operations support, and agile coaches. This grew to approximately 22 project members nine months into the project (Agile coach in the IT department).

Although most of the developers were hired consultants, the project was planned and managed by NAV employees. This differed from other projects, where software development and maintenance were outsourced to consultant companies through so-called responsibility contracts. The project was deemed a huge success, receiving the government's digitalization reward based on outstanding achievement. The chairman of the jury summarized the project as follows:

> This year's winners are a good example of how things can be done in new ways and deliver services that streamline public management and make life easier for users. Both the government and Difi [short in Norwegian for Agency for Public Management and eGovernment] have high expectations for digitalization of the public sector (Director at the Norwegian Digitalization Agency).

Following the success of the Digital Health project, NAV decided to implement the new delivery strategy across the organization. This required a series of changes: First, the agency changed the sourcing strategy, moving from outsourcing to insourcing of software development. As responsibility contracts expired, they were replaced by capacity contracts where consultants were hired by the hour.

> Development has been outsourced to consultant companies through traditional management contracts where the supplier has had independent responsibility for the system and maintenance. Now, NAV is taking over this. We have a different sourcing model where we hire people depending on our capacity instead of giving the supplier total responsibility (Agile coach in the IT department).

Second, the IT department was reorganized, replacing the hierarchical organization with a matrix structure. Whereas employees in the old structure were organized according to their role in the deployment cycle (planning, development, or operations), in the new structure, employees were grouped according to their competence field.

Third, software was developed by multidisciplinary teams that assumed responsibility for the entire software development cycle. In this way, development teams were able to learn from their actions and improve accordingly.

Fourth, the organization began to employ developers. Until this point, NAV had had few software developers of its own. Instead, the IT department was staffed to control suppliers and supervise projects. Development and maintenance of the NAV core systems had been outsourced to 15-20 different consultant companies, each pursuing its own goals and targets.

> We [NAV] had decided not to code ourselves. So, there we were, with 200-250 person-years, supervising 300-400 consultants. We were busy specifying needs and running tendering competitions and stuff like that, and sometimes, the needs proved too expensive. Then, we had to go back and clarify some more needs. So, there we were, managing and coordinating, being an intermediary [between the business side of NAV and the consultant company]. With roles that made make sure that the system was technically sound and things like that. But of course, it is quite difficult to control 15-20 suppliers, who all have their own interests and needs. How do you control them across 300 systems? How do you ensure the quality of the code base? Well, you can't. That's impossible (Manager in the IT department).

With the strategic shift, NAV gradually assumed responsibility for developing and maintaining the core systems. The goal was to increase the in-house development capacity and use consultants only during peak periods and to provide expert knowledge. The focus changed from controlling subcontractors to developing and maintaining the systems.

Second recoupling phase: Establishing product domains. As the applications became increasingly decoupled, the development teams could work more independently. However, many applications were part of larger value paths, and had to be developed in close relation to other systems and teams. A developer for the Parental Benefit project stated:

> In my experience, when people don't see the value chain they're contributing to, they begin to sub-optimize. It is artificial to say that you have a truly autonomous team in the area of Parental Benefit. All components and all applications support the same decision process.

To address dependencies, applications were grouped into functional domains, where a domain would contain several applications and teams.

> The idea behind domain-driven design is that it is more important to organize for flow. Optimal flow in the organization and in the code. And this is done by minimizing coordination and chattiness. You must bundle the things that belong together and keep the things that don't belong together apart. You bundle people and code in domains. People within a domain will be more closely linked (Manager in IT department).

By focusing on domains, NAV was able to establish value paths that transcended the organization. However, the domains had to be established gradually, one at a time. In this way, they could learn from the experiences from one domain before establishing the next.

> The goal is to remove boundaries between departments. But you can't do that by changing everything at once. All 19 000 employees. Because that won't work. Instead, you can do what we are doing. Establish one domain at a time. Gain experience and prove to the organization that it works. Then you establish the next domain (Senior executive in the Parental Benefit project).

As a principle, no legacy systems were brought into the domains. Domain teams would manage only new applications. In practice, NAV found that they had to make exceptions to this rule. In some cases, it was more practical to give domain teams responsibility for legacy systems while they were being dismantled and reimplemented.

By creating cross-cutting domains, NAV was able to improve flow and collaboration, without physically reorganizing employees. And by establishing one domain at a time, NAV could learn and adapt, gradually restructuring the organization and redesigning the service delivery. At the time of writing (September 2019), the organization had established three domains.

6 Discussion

The purpose of this study was to provide insight into how platformization unfolds within organizations. Platformization is often described as a process of establishing core services and an ecosystem of complementors (Benlian et al. 2018; Bygstad and Hanseth 2018; Cusumano 2010). However, we take a broader view, and draw on the concepts

Concepts	*Scope of theorizing*	*Manifestation in the case*
Decoupling	Decoupling legacy systems into a platform-oriented structure	Monolithic systems are gradually reimplemented as modular applications through application of microservices
	Decoupling information from its physical media (Liquefaction)	As systems are reimplemented, a growing number of manual operations are automated—liquefying data through the Kubernetes platform
Recoupling	Recombining knowledge and skills into new organizational forms and practices	Knowledge and skills are recombined through the formation of multidisciplinary teams
	Recombining applications and teams into domains	Applications and teams with stronger dependencies are recombined into product domains
Interaction of decoupling and recoupling	Cyclic interaction of decoupling and recoupling	Decoupling provides a stable platform for new combinations of knowledge and skills to emerge. In turn, recoupling provides new flexible organizational forms and practices for changing existing infrastructure into a platform architecture

Table 1. Platformization theorized.

of decoupling and recoupling to theorize how digital platforms enable new ways of organizing and delivering services in an interorganizational setting characterized by monolithic legacy systems and hierarchical structures. Empirically, we conducted a longitudinal case study of a platformization process in a Norwegian government organization, NAV, covering a period of seven years. Our detailed account of the platformization process in the case uncovers the way in which digital platforms pave the way for a reorganization of service delivery where outsourcing of development activities and a staged delivery model are replaced by continuous software development and cross-organizational collaboration. In the following, we draw on these empirical insights (as summarized in Table 4) and on extant literature to advance theory about platformization. In so doing, this paper provides three distinct contributions. Our first contribution is our theorization of how platformization involves processes of decoupling that can transform legacy systems into a modular platform architecture. In this way, decoupling does not merely provide a platform in addition to or on top of the existing digital infrastructure as presented in extant literature (Bygstad and Hanseth 2018; Islind et al. 2016), but instead, transforms the digital infrastructure into a working platform. Our second contribution is related to how platformization through processes of recoupling affords recombination of knowledge and skills into new organizational forms and practices. This gives an organization the capability to change infrastructures flexibly. The third contribution is our proposal that platformization requires cyclic interaction between processes of decoupling and recoupling in the sense that decoupling provides increased stability (i.e., a platform core) for new knowledge and skills to emerge, and in turn, recoupling implies increased flexibility and competence for change. Henceforth, the processes of decoupling and recoupling, although distinct, feed each other cyclically, so that if organizations have one without the other, they will not be able to implement platformization.

We begin by discussing the concept of decoupling before recoupling. Finally, we discuss how the concepts of decoupling and recoupling interact over time.

6.1 Decoupling: transforming legacy IS into platforms

By emphasizing decoupling as a strategy for renewing technology, we contribute to existing literature by suggesting that digital platforms provide a means for *replacing* legacy systems, thus addressing issues related to innovation along the periphery and at the core. Viewing platformization as a strategy for replacing legacy systems complements studies that see platformization either as a process of establishing platforms from scratch (Benlian et al. 2018; Islind et al. 2016) or as a process of masking legacy systems behind applications' programmable interfaces in an effort to increase innovation at the periphery (Bygstad and Hanseth 2018).

Decoupling corresponds to Islind et al.'s (2016) description of platformization as the process of establishing a digital platform. However, whereas Islind et al. describe platformization as the process of establishing a platform alongside existing infrastructure, we see decoupling as a process of establishing a platform structure across the existing infrastructure. Therefore, Islind et al. describe a case of green-field development, where the platform is established de neuvo—not having to relate to existing systems and practices. In contrast, the present study is of brown-field development, where the platform is tailored to the existing infrastructure. Our observations correspond to decoupling as described by Benlian et al. (2018), who identify it as unfolding on the infrastructure and application levels. Benlian et al. see decoupling as a general trend and a consequence of cloud computing. However, we take a more specific approach and see it as a strategy of technological renewal, enabling the gradual introduction of platform-oriented logic in existing infrastructures. This difference is important, because although the efficiency and transformative impact of platform structures have been proven through successful implementations, such as Apple (Eaton et al. 2015), large-scale organizations with a legacy of existing systems and practices are finding it hard to transition from tightly coupled architectures. In this vein, our conception of decoupling provides a strategy for gradually and continuously renewing technology which is applicable to even the most entrenched infrastructures. At NAV, decoupling progressed in two steps. In the first step, application systems were decoupled from their underlying digital resources through the use of virtual servers. Virtualization technologies provided increased elasticity as resources could be dynamically scaled up and down (Benlian et al. 2018). In the second step, legacy systems were decoupled into smaller applications. The container platform enabled reuse of third-party services and facilitated decoupling through the use of web protocols (i.e., SOAP and REST).

The use of platformization as a means for increased flexibility and innovation in digital infrastructures has been addressed by Bygstad and Hanseth (2018). In a multilevel study of a large e-health initiative, they examine the way in which a layered architecture, where legacy systems are encapsulated in the platform core, creates a platform-oriented infrastructure. Bygstad and Hanseth describe a process where legacy systems are encapsulated and hidden from application developers. However, we describe a process where legacy systems are gradually replaced. Rather than seeing these differing strategies as different and opposing, they should be seen as complementary and potentially mutually enabling, where the encapsulation of legacy systems might be seen as an aid, or a preliminary stage on the way toward renewing and replacing legacy systems. Although the approach proposed by Bygstad and Hanseth increases external innovation (offering complementors services which enable innovation), it leaves legacy systems mostly unchanged. Therefore, the strategy fails to improve the maintainability and evolvability of the legacy systems, which was seen as the main goal of the decoupling process as it unfolded at NAV. Decoupling legacy systems provided the agency with a set of modular components that could be combined and recombined into new products and services (Henfridsson et al. 2018), thus increasing resource density and the probability of innovation (Lusch and Nambisan 2015). The modular structure enabled the reallocation of applications to the most suitable actor in the organization (Normann 2001). In practice, with a large legacy of aging systems, it might not be realistic or desirable to replace all systems. Therefore, renewal be achieved through a combined strategy where some systems are hidden while others are replaced.

6.2 Recoupling: enabling new forms of organizing

In this study, we found that decoupling the legacy systems paved the way for an alternative method of organizing: Centralized control was replaced by a distributed structure where independent development teams were responsible for developing and managing applications. Teams were composed of technology experts and business domain representatives. This enabled innovation through the recombination of skills and knowledge across departments and subject domains (Brown and Duguid 1991). Thus, we propose that the process of recoupling allows for recombining knowledge and skills that enable flexible change in organizations.

At NAV, as most applications belonged to larger value chains, teams that were responsible for developing related functionality were combined into service domains. In this way, decoupling the digital infrastructure enabled the formation of a relatively self-contained, self-adjusting system of loosely coupled actors (Lusch and Nambisan 2015), where actors contributing to common value chains were more closely connected than actors contributing to different value chains (Evans 2006). Teams and domains transcended the matrix organization in a fluid structure, where employees could be dynamically combined and recombined in response to emergent needs (Ciborra 1996; Schreyögg and Sydow 2010).

These findings complement the platform literature by highlighting the way in which digital platforms enable restructuring of organizations through the dynamic recombination of teams and applications. By simultaneously opening the black boxes of technology and organization (Zammuto et al. 2007), we move beyond an inter-organizational perspective (Eaton et al. 2015; Yoo et al. 2010) and explore the way in which digital platforms facilitate innovation *within* organizations. From this intra-organizational view, digital platforms enable the recombination of not only digital resources (Henfridsson et al. 2018) but also knowledge and skills across sections and departments, further increasing the agency's potential for innovation.

The teams were composed of business and technology experts, providing the teams with the skills and knowledge to develop and manage applications independently. Combined with a modular platform where applications could be deployed in isolation, the teams were able to assume responsibility for all parts of the development process—from the inception of an idea until a service was eventually turned off. In this way, the organization was able to move from staged development, where different departments were responsible for different parts of the development process, to a continuous process in which one team was responsible for the entire software development cycle (Fitzgerald and Stol 2017). By shifting from the staged development practice to a continuous and network-oriented approach, NAV facilitated innovation across teams (Lusch and Nambisan 2015). These findings confirm insights from the software engineering literature where practices and continuous development are seen to enhance innovation through feedback and learning (Fitzgerald and Stol 2017). We complement these insights by exploring the way in which digital platforms can enable continuous development practices on a large scale. Practices of continuous development enabled feedback from users to be recombined into future versions of the service—thus adding an additional stakeholder in the recoupling process.

Although the concept of recombination is not new (Henfridsson et al. 2018; Lusch and Nambisan 2015), this research contributes by highlighting the interrelation between social organization and technical infrastructure, simultaneously opening the back boxes of technology and organization (Zammuto et al. 2007). As the teams are formed across existing structures, organizational recoupling is less prone to the knowledge disruption associated with structural recombination (Karim and Kaul 2015). However, at NAV, the transition from centralized to distributed control inferred a considerable shift in the power structures—where decision authority was transferred from centralized and coordinating roles to development teams. Therefore, the transition faced considerable resistance from parts of the organization, requiring coordinated efforts and persuasion at all levels of the organization. However, these issues are beyond the scope of this research.

6.3 Interaction between decoupling and recoupling

Grounded in this research, we theorize that processes of decoupling and recoupling interact cyclically, where decoupling increases the potential for recoupling and vice versa. Specifically, the analysis revealed that the process of decoupling provided new ways of organizing the development of information systems—thus increasing the organization's capacity to recouple. Similarly, the recoupling of the organization produced new organizational capabilities for renewing information systems, which, in turn, facilitated further decoupling (see Figure 2).

Consistent with previous research, this study showed that the decoupling of the digital infrastructure enabled a recombination of services (Benlian et al. 2018) and the introduction of an alternative organizational logic (Yoo et al. 2010). This research complements these studies by emphasizing platformization as an emergent phenomenon,

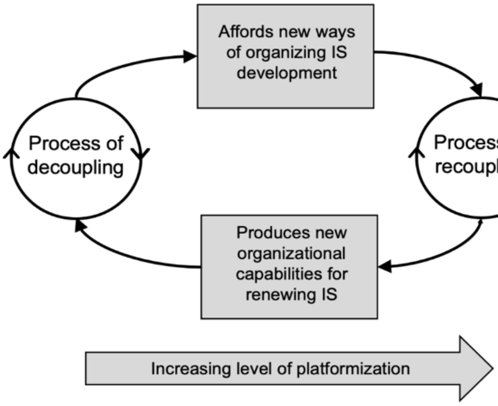
Affords new ways of organizing IS development
Process of decoupling
Process of recoupling
Produces new organizational capabilities for renewing IS
Increasing level of platformization

Figure 2. Interaction between decoupling and recoupling over time.

exploring the way in which the hierarchical organization is gradually and incrementally replaced by a distributed model.

NAV began by establishing one domain, and based on these experiences, continued to establish others. This incremental approach also faced resistance, as a proven track record (where independent development teams outperformed traditional project deliveries in terms of efficiency and flexibility) was a powerful and convincing argument in discussions with sceptics. In this manner, the incremental approach reduced resistance and increased the likelihood of a successful transition. With this incremental approach, the platformization process became a process of continuous organizational improvement, where feedback from one cycle was fed into subsequent cycles, allowing for a gradual and knowledge-based transformation of the organization.

7 Conclusion

This research presented a longitudinal study of the platformization of a large digital infrastructure. We found that through the decoupling of the digital infrastructure, the organization was able to dynamically recouple software components, skills, and knowledge into new and innovative services. We also found that the platformization unfolded cyclically, where the decoupling of applications enabled the recoupling of the organization, and the recoupling of the organization enabled further decoupling of the infrastructure.

As this research is based on a single case study, it is inevitably subject to several limitations. For example, the results cannot be validated across cases (Yin 2013). In addition, change processes of the magnitude described in this case study take a long time.

Although the transformation of NAV seemed successful, a two-year study is too short to draw any decisive conclusions. Thus, additional fieldwork is needed.

We observed the change process as it was unfolding—following events over a two-year period. Therefore, the findings suggest a path but are not conclusive regarding the outcome of the change process. Further studies are needed to provide detailed insights into the precise shortcomings and advantages of the decoupling process, where decoupling is investigated in a later stage of evolution.

Acknowledgment

We would like to thank NAV, and especially Petter Hafskjold and Ida Kristine Dørum for answering our questions and granting us access to the sites. We would also like to thank all informants for taking the time to talk to use. In addition, we want to acknowledge Eric Monteiro for providing extensive and valuable comments throughout the writing process.

This work was supported by the project Agile 2.0 supported by the Research Council of Norway through grant 236759 and by the companies DNV GL, Equinor, Kantega, Kongsberg Defence and Aerospace, Sopra Steria and Sticos.

References

Aanestad, M., and Jensen, T. B., (2011). Building nation-wide information infrastructures in healthcare through modular implementation strategies. *The Journal of Strategic Information Systems*, (20:2): 161-176.

Baldwin, C. Y., and Woodard, C. J., (2009). The architecture of platforms: A unified view. *Platforms, markets and innovation*): 19-44.

Battleson, D. A., West, B. C., Kim, J., Ramesh, B., and Robinson, P. S., (2016). Achieving dynamic capabilities with cloud computing: an empirical investigation. *European Journal of Information Systems*, (25:3): 209-230.

Benlian, A., Kettinger, W. J., Sunyaev, A., and Winkler, T. J., (2018). Special Section: The Transformative Value of Cloud Computing: A Decoupling, Platformization, and Recombination Theoretical Framework. *Journal of Management Information Systems*, (35:3): 719-739.

Boudreau, K., (2010). Open Platform Strategies and Innovation: Granting Access vs. Devolving Control. *Management Science*, (56:10): 1849-1872.

Brown, J. S., and Duguid, P., (1991). Organizational learning and communities-of-practice: Toward a unified view of working, learning, and innovation. *Organization science*, (2:1): 40-57.

Bygstad, B., and Hanseth, O., (2018). TRANSFORMING DIGITAL INFRASTRUCTURES THROUGH PLATFORMIZATION.).

Cennamo, C., and Santaló, J., (2019). Generativity tension and value creation in platform ecosystems. *Organization Science*, (30:3): 617-641.

Charmaz, K., (2014). *Constructing grounded theory*, Sage.

Ciborra, C., (2000). *From control to drift: the dynamics of corporate information infastructures*, Oxford University Press on Demand.

Ciborra, C. U., (1996). The platform organization: Recombining strategies, structures, and surprises. *Organization science*, (7:2): 103-118.

Constantinides, P., Henfridsson, O., and Parker, G. G. (2018). Introduction—Platforms and Infrastructures in the Digital Age: INFORMS.

Cusumano, M., (2010). Cloud computing and SaaS as new computing platforms. *Communications of the ACM*, (53:4): 27-29.

de Reuver, M., Sørensen, C., and Basole, R. C., (2017). The digital platform: a research agenda. *Journal of Information Technology*).

Eaton, B., Elaluf-Calderwood, S., Sorensen, C., and Yoo, Y., (2015). Distributed tuning of boundary resources: the case of Apple's iOS service system. *MIS Quarterly: Management Information Systems*, (39:1): 217-243.

Eisenhardt, K. M., (1989). Building theories from case study research. *Academy of management review*, (14:4): 532-550.

Evans, E., (2004). *Domain-driven design: tackling complexity in the heart of software*, Addison-Wesley Professional.

Fitzgerald, B., and Stol, K.-J., (2017). Continuous software engineering: A roadmap and agenda. *Journal of Systems and Software*, (123): 176-189.

Gawer, A., (2014). Bridging differing perspectives on technological platforms: Toward an integrative framework. *Research Policy*, (43:7): 1239-1249.

Gawer, A., and Cusumano, M. A., (2014). Industry platforms and ecosystem innovation. *Journal of Product Innovation Management*, (31:3): 417-433.

Ghazawneh, A., and Henfridsson, O., (2013). Balancing platform control and external contribution in third-party development: the boundary resources model. *Information Systems Journal*, (23:2): 173-192.

Hanseth, O., (2001). Gateways—just as important as standards: How the internet won the "religious war" over standards in Scandinavia. *Knowledge, technology & policy*, (14:3): 71-89.

Hanseth, O., and Lyytinen, K., (2010). Design theory for dynamic complexity in information infrastructures: the case of building internet. *Journal of Information Technology*, (25:1): 1-19.

Helmond, A., (2015). The platformization of the web: Making web data platform ready. *Social Media+ Society*, (1:2): 2056305115603080.

Henfridsson, O., and Bygstad, B., (2013). The generative mechanisms of digital infrastructure evolution. *MIS quarterly*, (37:3).

Henfridsson, O., Nandhakumar, J., Scarbrough, H., and Panourgias, N., (2018). Recombination in the open-ended value landscape of digital innovation. *Information and Organization*, (28:2): 89-100.

Islind, A. S., Lindroth, T., Snis, U. L., and Sørensen, C., (2016). Co-creation and fine-tuning of boundary resources in small-scale platformization, *Scandinavian conference on information systems*, Springer, pp. 149-162.

Jacobides, M. G., Cennamo, C., and Gawer, A., (2018). Towards a theory of ecosystems. *Strategic Management Journal*).

Kallinikos, J., Aaltonen, A., and Marton, A., (2013). The ambivalent ontology of digital artifacts. *Mis Quarterly*): 357-370.

Karim, S., and Kaul, A., (2015). Structural recombination and innovation: Unlocking intraorganizational knowledge synergy through structural change. *Organization Science*, (26:2): 439-455.

Kornberger, M., Pflueger, D., and Mouritsen, J., (2017). Evaluative infrastructures: Accounting for platform organization. *Accounting, Organizations and Society*, (60): 79-95.

Lusch, R. F., and Nambisan, S., (2015). Service innovation: A service-dominant logic perspective. *MIS quarterly*, (39:1).

Normann, R., (2001). *Reframing business: When the map changes the landscape*, John Wiley & Sons.

Parker, G. G., Van Alstyne, M. W., and Choudary, S. P., (2016). *Platform Revolution: How Networked Markets Are Transforming the Economyand How to Make Them Work for You*, WW Norton & Company.

Poell, T., Nieborg, D., and van Dijck, J., (2019). Platformisation. *Internet Policy Review*, (8:4): 1-13.

Rolland, K. H. M., Lars; Rai, Arun, (2018). Managing digital platforms in user organizations: The interactions between digital options and digital debt. *Information Systems Research*, (29:2): 419-443.

Schreyögg, G., and Sydow, J., (2010). Crossroads—organizing for fluidity? Dilemmas of new organizational forms. *Organization science*, (21:6): 1251-1262.

Spagnoletti, P., Resca, A., and Lee, G., (2015). A design theory for digital platforms supporting online communities: a multiple case study. *Journal of Information technology*, (30:4): 364-380.

Tan, H., and Wright, J., (2018). A Price Theory of Multi-Sided Platforms: Comment. *American Economic Review*, (108:9): 2758-2760.

Tiwana, A., (2013). *Platform ecosystems: aligning architecture, governance, and strategy*, Newnes.

Tiwana, A., Konsynski, B., and Bush, A. A., (2010). Research commentary—Platform evolution: Coevolution of platform architecture, governance, and environmental dynamics. *Information Systems Research*, (21:4): 675-687.

Törmer, R. L., and Henningsson, S., (2018). From Drift To Central Guidance: A Path Constitution Perspective On The Platformization Of An Information Infrastructure, *2018 European Conference on Information Systems*, Association for Information Systems. AIS Electronic Library (AISeL).

Vassilakopoulou, P., Grisot, M., Blegind Jensen, T., Sellberg, N., Eltes, J., Thorseng, A. A., and Aanestad, M., (2017). Building national ehealth platforms: the challenge of inclusiveness, *Proceedings of the International Conference on Information Systems-Transforming Society with Digital Innovation, ICIS 2017*.

Wareham, J., Fox, P. B., and Cano Giner, J. L., (2014). Technology ecosystem governance. *Organization Science*, (25:4): 1195-1215.

Yin, R. K., (2013). *Case study research: Design and methods*, Sage publications.

Yoo, Y., Boland Jr, R. J., Lyytinen, K., and Majchrzak, A., (2012). Organizing for innovation in the digitized world. *Organization science*, (23:5): 1398-1408.

Yoo, Y., Henfridsson, O., and Lyytinen, K., (2010). Research commentary—the new organizing logic of digital innovation: an agenda for information systems research. *Information systems research*, (21:4): 724-735.

Zammuto, R. F., Griffith, T. L., Majchrzak, A., Dougherty, D. J., and Faraj, S., (2007). Information technology and the changing fabric of organization. *Organization science*, (18:5): 749-762.